For Submission to the SPIN Workshop 2001, Toronto, Canada on January 29, 2001
http://www.cis.ksu.edu/santos/spin2001/,

Forecasting Model Checking Needs during Early Requirements Engineering with Rapture/SP2

John D. Powell[1], Tim Menzies[2]

[1] [1]Jet Propulsion Laboratory, 4800 Oak Grove Drive, Pasadena CA 911098099,
john.powell@jpl.nasa.gov
[2]Dept. Electrical & Computer Eng., University of British C o-
lumbia, 2356 Main Mall, Vancouver, B.C., Canada, V6T 1Z4,
tim@menzies.com

Abstract. Effectively forecasting future model checking needs during the concept and early requirements phases of software can facilitate appropriate application of this technology to maximize its overall benefit. The effort associated with model checking may recoup itself and net overall value added by resulting in the timely discovery of hazardous critical system software behavior that may otherwise go undiscovered. However, in order to achieve this net benefit from model checking, the technology must be applied to appropriate "small" portions of the system that are likely to produce such behavior. The Rapture/SP2 alternative to traditional model checking allows less complex requirements often seen in the concept stage to be modeled in the form of topi diagrams. Within the expressive limitations of this method, many properties of interest may be examined over large system models with a high (>=90%), degree of confidence. The "large portions" of the system(s) represented are equivalent to state spaces that are beyond the reasonable capabilities of state of the art model checkers like SPIN. The proposed usage of Rapture/SP2 in this paper is to identify "smaller" portions of a system that are more likely to contribute to hazardous behavior. This provides objective justification for future (traditional) model checking efforts to be focused in the areas of the system that are most likely to yield critical errors and anomalies.

1 Introduction

During early the concept and design of new systems general relationships between portions of the system are readily identified [1]. The relationships may form early requirements that are vague and volatile. While traditional model checking, such as the use of SPIN [2,3,4,5] may be applied to these requirements, Rapture/SP2 may offer a more viable approach until early requirements become more stable and specific. The volatile nature of these requirements can prove disruptive to traditional model checking efforts by requiring SPIN model(s) to be altered often. This results in difficulties in maintaining fidelity between a specific SPIN model and the rapidly evolving system, especially evolution that causes growth in system.

The manner in which system evolution takes place in early the stages of development is important. When initial relationships are examined, they are expanded as new relationships are discovered. The progression towards specification of system behavior over a growing base of functionality quickly results in an explosion of the associated model checking state space. Systematic evaluation of properties via modeling checkers such as SPIN over the system concept as a whole becomes unviable.

Rature/SP2 allows verification of temporal properties while keeping the system concept in tact as it evolves. This is achieved via trade-offs in expressability and the level of confidence in Rapture/SP2 verification results. Rature/SP2's confidence level that a given model does not violate a property is not absolute as it is in SPIN. However, property violations that are found in the model with Rapture/SP2 retain 100% confidence.

Confidence Levels		
Verification Result	SPIN	Rapture/SP2
P Holds for Model	100%	>90%
P Violated in Model	100%	100%

By tracking violations reported by Rapture/SP2 over the system at large as it evolves early in the life cycle, objective evidence that certain functionalities contribute to more dangerous interactions than others can be ascertained. Subsequently, an evaluation of the severity associated with these dangers components can be performed. Finally, traditional model checkers such as SPIN can be applied to the smaller portions of the system that are responsible the "most" severe interactions.

This approach allows Rapture/SP2 and SPIN to compliment each other. In addition to providing objective justification of the need for appropriate future model checking, the logical segmentation of the system will be more clearly defined to ensure maximum value added through verification of the existence/absence of critical anomalies in a SPIN model. Rapture/SP2 verifies properties system wide during the concept and early requirements phases. Through early awareness of erroneous, conflicting or inadequate early requirements derived from the concept, many anomalies can be avoided inexpensively by adjusting the concept design. SPIN is then used to verify a subset of the Rapture/SP2 properties with an absolute level of confidence. That subset represents those properties that must hold to avoid catastrophic system failures.

2 Rapture/SP2 Functionality

Rature/SP2 combines the SP2 algorithm and a set of heuristic rules. The SP2 algorithm is a variant of Djkstra's shortest path algorithm [7,8] that efficiently traverses symmetric topi and partitions their nodes into two time partitions S' and T' [1].

Nodes are placed in the S' time partition as they are reached until a node is reached that conflicts with a member of S'. The conflicting node and all nodes reached in the resulting downstream path are placed in T'. This allows the system's nodes (events) to be viewed in terms of two points in time now (S') and later (T'). The SP2 algorithm has been shown to run in O(|V|+|E|log|V|) time [1,6].

The heuristic reasoning component of Rature/SP2 examines characteristics of the shortest path tree produced by SP2. It corrects temporal inconsistencies associated with and-node semantics in the tree by adjusting edge weight in a manner that causes a subsequent run of the SP2 algorithm to pursue new paths in the topi that avoid the inconsistencies [1,9].

Experimentation with Rature/SP2 demonstrated its ability to efficiently evaluate temporal properties over system spaces that are equivalent to state spaces sizes as high as $2^{10,000}$ [6].

Rapture SP2 Limitation Details

The confidence rating depends on:
1. The largest reachable portion of the systems space achieved.
2. The probability that further heuristic correction will increase the system space coverage to 100%
3. The probability that a yet-to-be-examined shortest path tree with 100% coverage of the system space exists

If a property still holds after n heuristics corrections are performed without 100% coverage of the system space, the confidence rating is <100% because the n+1 heuristic correction could yield a higher coverage of the system space containing the violation. When 100% coverage of the system space is achieved without finding a property violation the confidence rating is <100% to the degree that multiple shortest path trees with different temporal configurations of the nodes exist. One of the alternate configurations may represent legal behavior that violates the property.

The limitation on expressability is two fold. First is the limitation on the types of models that may be expressed. The characteristics of the symmetric topi that SP2 may examine preclude expressing complex functionality such as iteration, recursion and subroutine calls [1]. Secondly, a substantial, but not complete, set of temporal *occurrence* properties can be expressed and a smaller set of *ordering* properties is expressible [1,6,10,11]. However, within the scope of expressible properties are many properties of interest pertaining to common early concept / requirements issues. This property expression limitation is due to the fact that only two distinct points of time are distinguishable in Rapture/SP2. Thus ordering properties that express three or more properties in a given order such as nested until (U) operators in linear temporal logic cannot be expressed [1,6,10,11].

3 Rapture/SP2 and SPIN Compliment Approach

The approach of using Rapture/SP2 as a complimentary precursor to the use of SPIN facilitates fast, inexpensive, early correction of concept level conflicts (See Figure 1). By verifying that some properties are violated by the specification before SPIN is even employed, early correction can be made. Thus the system space that needs to be modeled in SPIN may be reduced. The remaining properties that possibly hold are evaluated for severity to form the "Severe Risk Subset". At this point only portions of the system behavior relevant to the "Severe Risk Subset" need to be modeled for SPIN. The model size is not only potentially reduced but the portion of the system being modeled can be justified objectively via the data from the Rature/SP2 results. Finally, the "Severe Risks Anomalies..." that survive the early concept / early requirements phases are reduced.

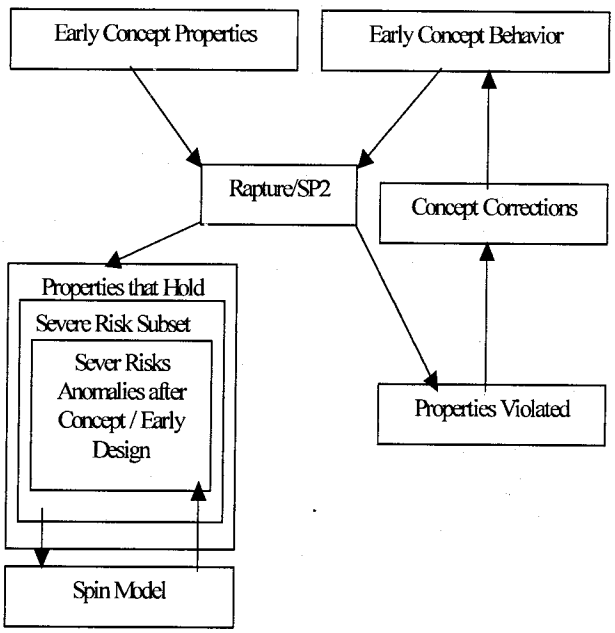

Figure 1: Rapture/SP2 – SPIN Approach

The behavior of the relationships in the concept is translated in to a series of edges in the topi. This process is less complex at this stage due to the intuitive nature of the relationships. For example, some relationships may be:

- An increase in engine thrust *leads to* an increase in airspeed.
- Decrease in ground speed *leads to* increased trip time

Properties may be translated from linear temporal logic formulae that to not explicitly express more than to distinct points on time to S' and T' membership criteria via a systematic grammar. Each node in the topi represents a system event (i.e. increase of X, decrease of Z, etc...). The phrase *"leads to"* in the examples above implies edges in the topi. When Rapture/SP2 is executed the nodes are assigned to either the S' or T' set. Verifying a property is merely a membership test of for appropriate nodes in each set. This method of property verification allows multiple properties to be verified over a given topi representation of the system with a given confidence level after only one execution of SP2.

While not currently implement this feature can be easily automated. Although the confidence level will not be the optimal one for each property this represents a significant economy of scale when a sufficient confidence level can be established in advance. This may be view as a form of concurrent property verification. By utilizing concurrent results properties may be examined in combination with minimal (insignificant) additional processing overhead.

4 Conclusion

Rapture/SP2's present value as a precursory compliment to SPIN far exceeds is current value as a replacement. This is particularly true for critical systems where catastrophic failure results in major loss of assets or loss of life. Model checking, as well as other formal methods approaches, remain necessary to the assurance efforts of mission critical software. However, Rapture/SP2 can offer a valuable approach to early detections of requirements deficiencies. When Rapture/SP2 is used in conjunction with the SPIN model checker the approach can offer many benefits including:

- Larger system models (topi) may be examined in a formal, although somewhat less rigorous manner.
- Expensive Model Checking resources (Model Checking experts, domain experts,) need not be employed to identify many concept / early requirements problems.
- Many problems identified by Rapture/SP2 in the earliest phases of the life cycle may be inexpensively fixed before they proliferate into more formal and detailed requirements.
- The model checking effort is more efficient because specific objective information about reduced set potentially high-risk problems is available before SPIN is employed. This allows modeling to focus on a potentially smaller portion of the system with reduced probability that significant errors will be found.

References

1. Menzies, Tim., Powell, J., Houle, M.: Fast Formal Analysis of Requirements via "Topi Diagrams". ICSE 2001, Toronto, Canada
2. Holzmann, G.: The SPIN Model Checker. IEEE Transaction on Software Engineering. 23(5):279-295, May 1997
3. Holzmann, G., Puri, A.: A Minimized Automaton Representation of Reachable States. Software Tools for Technology Transfer. Springer Verlag 1999
1. Holzmann, G.: Design and Validation of Computer Protocols. Prentice Hall 1990 ISBN: 0135399254 .
2. Holzmann, G.: The Promela Reference. http://cm.bell-labs.com/cm/cs/what/spin/Man/full.html
3. Powell, J.: A Graph theoretic Approach to Assessing Tradeoffs on Memory Usage for Model Checking. Thesis West Virginia University, 2000
4. Dijkstra, E.: A Note on Two Problems in Connection with Graphs. Numerische Mathmatik, 1:269-271, 1959
5. Cormen T., Leiserson, C., Rivest R.: Introduction to Algorithms MIT Press, 1990 ISBN: 0262031418
6. Menzies, T., Houle, M., Powell J.: Rapture/SP2: Efficient Testing of Temporal Properties without Search Space Explosion, 1999. NASA IV&V Technical Report,http:// research.ivv.nasa.gov/docs/techreports.html.
7. Dwyer, M. Avrunin, G., Corbett J.: Patterns in Property Specifications for Finite-State Verification. ICSE98: Proceedings of the 21st International Conference on Software Engineering, May 1998
8. Dwyer, M. Avrunin, G., Corbett J.: A System Specification of Patterns. http://www.cis.ksu.edu/santos/spec-patterns/, 1997.